24x7 Monitoring and Troubleshooting Distributed Application Performance

WHITE PAPER

Gone is the luxury of sending network engineers to physically visit a site to troubleshoot performance issues. Today's geographically distributed companies and distributed applications require a 24x7 proactive approach to measuring and monitoring application performance. This paper defines Application Response Time (ART), presents factors to consider when troubleshooting the performance of distributed applications, and identifies "must have" criteria for distributed application analysis solutions.

Ihr Kontakt:

DATAKOM
Networld Intelligence

DATAKOM Ges. für Datenkommunikation mbH
Lise-Meitner-Str. 1 · 85737 Ismaning
Tel. +49 89 996525-10 · info@datakom.de

WildPackets, Inc.
1340 Treat Blvd, Suite 500
Walnut Creek, CA 94597
925.937.3200
www.wildpackets.com

24x7 Monitoring and Troubleshooting Distributed Application Performance

Introduction

Distributed networks, which include pretty much any corporate network today, require distributed analysis, the collection of network data across multiple key points in the network, 24/7. These networks typically run multiple applications, from single-tier, locally-hosted applications like e-mail, to multi-tier web-based applications, or even time-sensitive, multi-hop applications like VoIP. While application traffic has historically resided within the Data Center, SaaS and cloud computing are driving application traffic outside of the traditional enterprise network, making network latency even more of an issue.

This model is in stark contrast with the portable analysis approach where you capture data only after a problem has been reported, and do so by moving around to different points in the network with a laptop or other mobile device running network analysis software. With today’s high speed and highly distributed networks, this approach is often too little too late, though it does remain a viable option in some smaller wireless LAN (WLAN) infrastructures.

Keeping up the performance of your applications is essential to your business, and understanding how to monitor network and transaction performance, the two basic components of application performance, will help you keep your users happy and your network healthy. Pinpointing and correcting slowdowns is therefore a necessity, and can be a real challenge.

To pinpoint what is responsible for the latency creating poor application performance – the network or the application itself – you need to measure and monitor application response time (ART). This white paper defines ART and presents factors to consider when troubleshooting the performance of distributed applications.

Defining Application Response Time

Networks are like driving down the highway: every driver is different when it comes to the speed at which they feel comfortable driving. Weather conditions, mood, or simply time of day can cause the driver to adjust their speed. The same is true with applications and the time users are willing to wait for them to respond. Some users don’t mind a long wait time to download a report, but they will feel frustrated when a usually instantaneous application is taking more than a few seconds to respond.

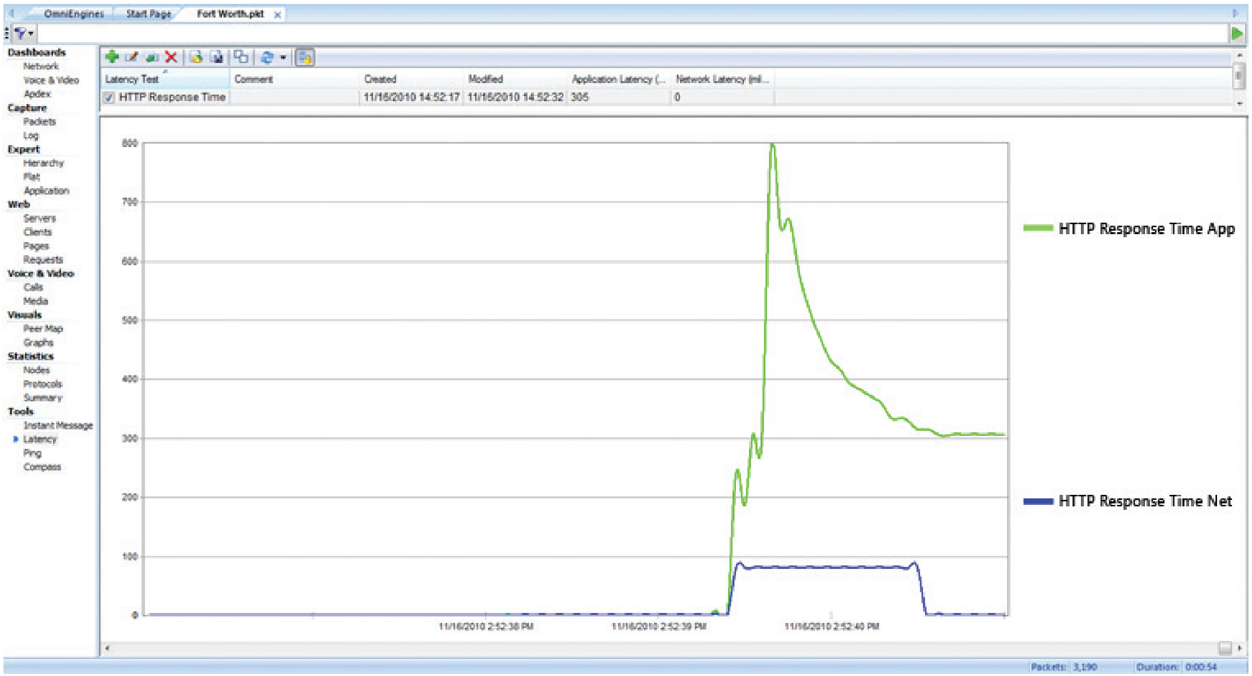
Ensuring that your network and applications are performing optimally is essential for your business. Application performance relates to the time it takes an application to respond to a specific user request, measured from the user’s perspective, through either the network and/or the web services infrastructure. ART is really about user experience, and although each user may have a different impression of acceptable performance, the ways in which to properly measure overall performance are well established.

Latency

First, we must distinguish between the two basic types of latency – network and application latency. *Network latency* is the amount of time it takes for an application request or transaction to travel from the client to the server and back over the network, sometimes referred to as the round trip transaction time. *Application latency* is the amount of time needed for the application on the server to process the request and send a response containing real data.

Clearly measuring network latency vs. application latency is the proof the network engineer needs. Latency monitoring can help correlate areas of latency with other relevant statistics, as well as the actual network traffic occurring at that time. This type of high-resolution forensic analysis can help to detect latency problems at the highest level and drill down quickly for closer inspection.

Packet-level monitoring is ideal for accumulating evidence. By visually inspecting a packet-level conversation between a client and a poorly performing application, one can see whether the network (or a network device) is the source of the delay, or if the bottleneck is the application. This is done by comparing the responsiveness of the TCP ACK to a client request versus the application response in the TCP payload. Quite often the server's network stack acknowledges the client request quickly (within milliseconds), while the application may take tenths of seconds or even seconds to respond with payload data. When you see this, you know it's the application causing the problem. Conversely, if the application is fast, the payload may be included in the initial ACK, implying that the latency is coming from the network.



Latency Test	Comment	Created	Modified	Application Latency (...	Network Latency (mil...
☑ HTTP Response Time		11/16/2010 14:52:17	11/16/2010 14:52:32	305	0

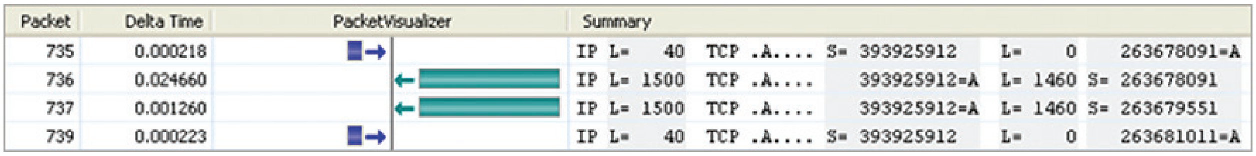
Packets: 3,190 Duration: 0:00:54

Ideally, network latency and application latency measurements can be graphed together over time. Comparing the measurements and seeing the differences can provide information that might have otherwise been overlooked, and aid in identifying transient application problems.

Calculating and Measuring Application Response Time

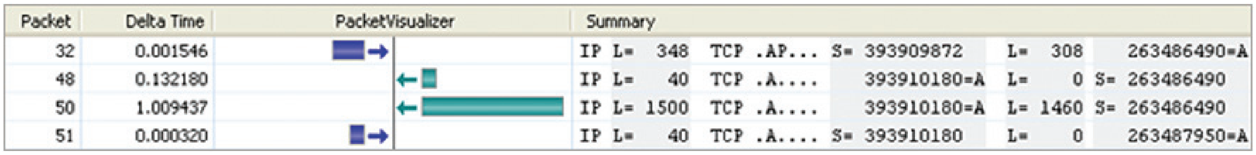
Application performance depends heavily on network performance, so if your network is not performing correctly, then your users will experience problems with their applications. If an application is not performing properly, users typically blame the network as the culprit for the issue. If a problem arises and users become frustrated with the performance of an application, which is really the only perspective that an end-user has, the first step is figuring out what is causing the problem: the network or the application.

Possible Issues on the Network



Packet	Delta Time	PacketVisualizer	Summary
735	0.000218	→	IP L= 40 TCP .A.... S= 393925912 L= 0 263678091=A
736	0.024660	←	IP L= 1500 TCP .A.... 393925912=A L= 1460 S= 263678091
737	0.001260	←	IP L= 1500 TCP .A.... 393925912=A L= 1460 S= 263679551
739	0.000223	→	IP L= 40 TCP .A.... S= 393925912 L= 0 263681011=A

Possible Issues with the Application



Packet	Delta Time	PacketVisualizer	Summary
32	0.001546	→	IP L= 348 TCP .AP... S= 393909872 L= 308 263486490=A
48	0.132180	←	IP L= 40 TCP .A.... 393910180=A L= 0 S= 263486490
50	1.009437	←	IP L= 1500 TCP .A.... 393910180=A L= 1460 S= 263486490
51	0.000320	→	IP L= 40 TCP .A.... S= 393910180 L= 0 263487950=A

ART is broken down by Peter Sevcik and Rebecca Wetzel of NetForecast into two key components:

- *The network response time (NRT),* which addresses just the network latency, or the time it takes data (a packet or set of packets) to traverse the network from the end user to the network location of the application processing hardware (virtual or otherwise) and back
- *The transaction response time (TRT),* which addresses the processing time required by all application processes.

The following sections break down each of these components further leveraging the work of NetForecast.

Network Response Time

The **Network Response Time (NRT)** is the time between a user's action and the network's response to that action. Be aware that this component is dynamic, never a constant, as a number of elements factor into it, including the payload size, overall network bandwidth available to the user/application, the maximum transmission unit (MTU) along the data path, and the round trip response time (RTT) – the time it takes data packets to simply traverse the network including any switching or routing latencies.

Network Response Time (NRT) is approximately
(Payload/Bandwidth) + [AppTurns*RTT] where

- **Payload:** Information content in bytes
- **Bandwidth:** Minimal link speed between client and server
- **AppTurns:** Number of interactions needed between the client and the server to provide a response to the user
- **Round Trip Response Time (RTT):** Propagation time for data between the client and the server

As this equation shows, though many aspects of network response time are within a network engineer's control, some aspects are still application dependent, including the overall payload sizes and the number of application turns. AppTurns are especially worrisome, as each turn incrementally adds the RTT to the total NRT delay, possibly making the network look slow even though it's really poor application design. A common example of an AppTurn violator is client-side filtering, where the server returns the entire dataset to the client for every query, saving server resources at the expense of the client and the network!

Network bandwidth includes not only link speeds, but also factors such as congestion and MTU. A crowded link can cause resource contention and packet queuing, adding latency to an otherwise fast connection. Extreme cases may result in dropped packets, requiring retransmissions, which add network latency that appears similar to the amount from an AppTurn. Dropped packets are easy to spot with session-oriented protocols like TCP, because there will be oddities in the sequence numbers. Only slightly less worrisome are fragmented packets, in which a packet larger than the MTU on a link is broken into multiple packets. The number of packets goes up, and the apparent bandwidth goes down. While fragmentation used to be caused by low MTU settings on WAN links, the typical cause today is unaccounted overhead on tunneling protocols, like IPSec. For efficiency, packets are tuned to be as large as possible, so adding another header will make the packet too big for the wire, ironically causing inefficiency. Fragments are also easy to see, because a large packet will be followed immediately by a small packet. IP also includes header flags to indicate that fragmenting happened. Since both of these effects artificially lower bandwidth, it's good that they're easy to spot.

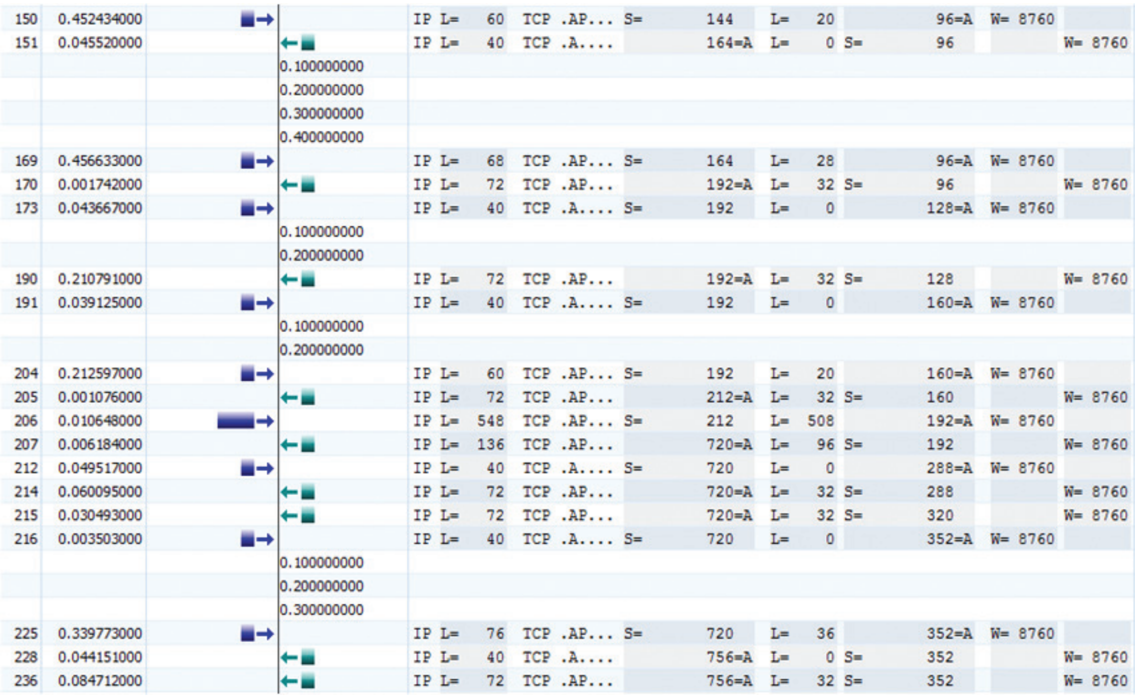
When troubleshooting network response time, remember that there are other connection elements besides just the client and the server. With proxy servers, the TCP connection is from client to proxy, and from proxy to server, not end-to-end. In multi-tier application environments, you must also remember to calculate the network latency between each tier. Also, be sure to check DNS response time as that is a common source of performance issues, especially in web-based applications. DNS issues often appear to users as a long wait only the first time they access the server. A packet-level analysis will be very clear if the delay is coming from the DNS lookup, as the DNS connection will have a long response time, but the actual server connection will not. However, it's easy to accidentally exclude DNS from a packet capture if the capture filters don't include the DNS server address.

Transaction Response Time

The **Transaction Response Time (TRT)** is the amount of time the application must spend processing a request before it returns some sort of response on the network, data or otherwise.

Transaction Response Time (TRT) is approximately
Server Response Time (SRT) + Client Response Time (CRT) where

- **Server Response Time (SRT):** Processing time required by the server
- **Client Response Time (CRT):** Processing time required by the client



```
150  0.452434000   ■→               IP L=  60  TCP .AP... S=  144     L=  20          96=A  W= 8760
151  0.045520000        ←■          IP L=  40  TCP .A....     164=A   L=   0  S=  96            W= 8760
                        0.100000000
                        0.200000000
                        0.300000000
                        0.400000000
169  0.456633000   ■→               IP L=  68  TCP .AP... S=  164     L=  28          96=A  W= 8760
170  0.001742000        ←■          IP L=  72  TCP .AP...     192=A   L=  32  S=  96            W= 8760
173  0.043667000   ■→               IP L=  40  TCP .A.... S=  192     L=   0         128=A  W= 8760
                        0.100000000
                        0.200000000
190  0.210791000        ←■          IP L=  72  TCP .AP...     192=A   L=  32  S= 128            W= 8760
191  0.039125000   ■→               IP L=  40  TCP .A.... S=  192     L=   0         160=A  W= 8760
                        0.100000000
                        0.200000000
204  0.212597000   ■→               IP L=  60  TCP .AP... S=  192     L=  20         160=A  W= 8760
205  0.001076000        ←■          IP L=  72  TCP .AP...     212=A   L=  32  S= 160            W= 8760
206  0.010648000  ■■■→              IP L= 548  TCP .AP... S=  212     L= 508         192=A  W= 8760
207  0.006184000        ←■          IP L= 136  TCP .AP...     720=A   L=  96  S= 192            W= 8760
212  0.049517000   ■→               IP L=  40  TCP .A.... S=  720     L=   0         288=A  W= 8760
214  0.060095000        ←■          IP L=  72  TCP .AP...     720=A   L=  32  S= 288            W= 8760
215  0.030493000        ←■          IP L=  72  TCP .AP...     720=A   L=  32  S= 320            W= 8760
216  0.003503000   ■→               IP L=  40  TCP .A.... S=  720     L=   0         352=A  W= 8760
                        0.100000000
                        0.200000000
                        0.300000000
225  0.339773000   ■→               IP L=  76  TCP .AP... S=  720     L=  36         352=A  W= 8760
228  0.044151000        ←■          IP L=  40  TCP .A....     756=A   L=   0  S= 352            W= 8760
236  0.084712000        ←■          IP L=  72  TCP .AP...     756=A   L=  32  S= 352            W= 8760
```

When troubleshooting transaction response time, there are various factors to consider. On the server side, SRT is impacted by:

- **Server Latency:** Time for the server to process a request. This latency is dependent upon memory, disk system, CPU, and usage.
- **Application Latency:** Time for the application itself to process the request. This latency is mostly software design dependent.
- **Back End Latency:** If there are multiple tiers in the design, like middleware or a database, this latency is affected by the database design, fragmentation, indexing, etc.

On the client side, **browser/workstation latency** is typically the biggest contributor.

Within the TRT, the server response time (SRT) is usually the dominant element, but client response time (CRT) can also be a significant factor if the application relies significantly on client-side processing (such as Flash or Java applets).

Protocol Suitability

Independent of both NRT and TRT is protocol suitability to the application. A lot of network applications, especially those designed and used “in-house,” are built with high-level toolkits and APIs which hide the underlying details from the developer. This layer of abstraction can hide classic problems from the developer, like packet fragmentation: “large” packets may be efficient in a small test LAN, but can be split into 2 packets due to encapsulation (like MPLS or

IPSec) in deployment. Rapid release schedules leave little time for testing and tuning in the real world environment, leading to "it worked in the lab!" conversations.

Understanding Available Network Analysis Architectures

Nobody's network is the same, and the topology depends on many factors, but most networks have similar characteristics which can be used to help you plan for a holistic solution that will best monitor and analyze your entire distributed environment.

As the above definitions clearly identify, we really need to see an overall application "in action" – packet by packet – to truly understand the overall response and the individual contributions from the network and the transactional elements of the application. Packet-level network analysis is the best way to get the data you need. Packet analysis can display data flow by flow, making it very easy to isolate the communication between a single user and a single application. By doing this you are able to organize your analysis into separate flows to see where potential issues are and address the issue of network vs. application.

To collect relevant data, you need to decide where to monitor. It is common to start by capturing at a single point, and choose between server- or client-side monitoring. Client-side monitoring is great to get the most accurate picture of what the client sees, but the latency measurements will include both the Server Response and the Network Response, which obscures the information about where the problem is if there's not an obvious gap between ACK and data in the server responses. It also requires monitoring equipment at all client sites, even remote sites, which can drive up the cost.

Most enterprises employ server-side monitoring, which can be less expensive and better able to assess server response time, but it hides the Network Response, making measurements to differentiate individual client locations more difficult.

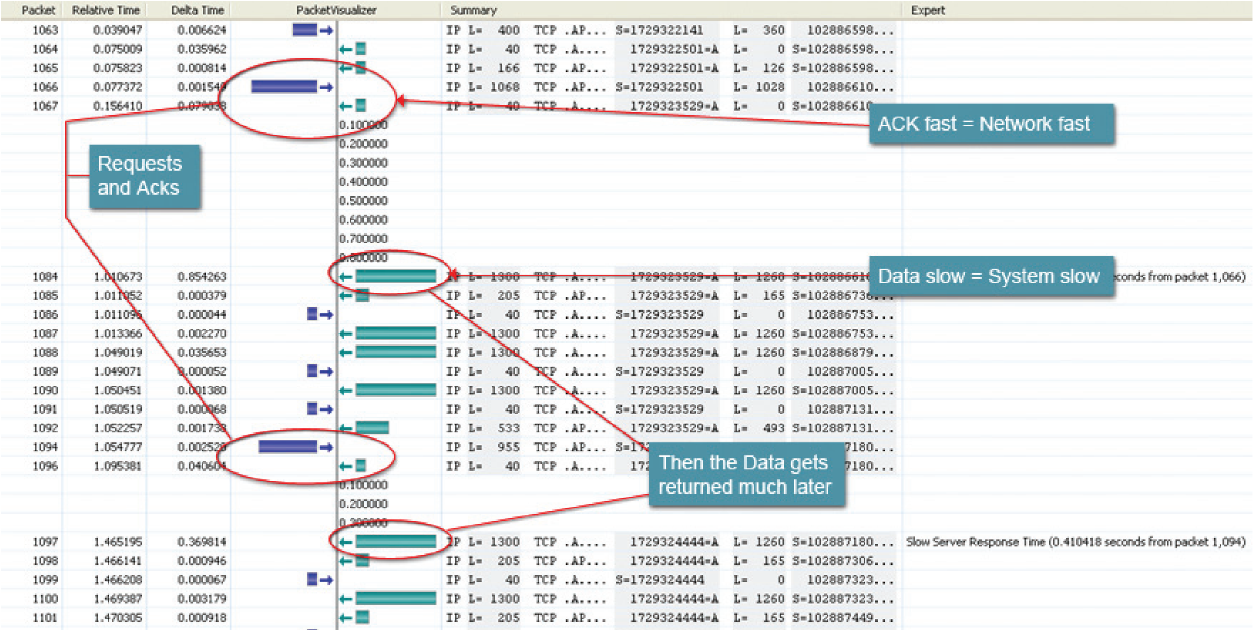
The most useful data comes from networking monitoring on both the client and the server. Measuring at the server provides Server Response time. Measuring at the client includes both Server and Network response times.

Benefits of Employing Distributed, 24x7 Solutions

Ongoing monitoring can provide insight into network performance on key interfaces, and can alert you when conditions begin to decline. Before you can tell if network and transaction latency are snowballing, you must have an understanding of what normal means for your network.

Benchmarking your application response time provides you with details of how your applications regularly perform. For example, let's look at the example of a web-hosted CRM application. If you run periodic checks in the background to the application host, it can provide an ongoing baseline of the performance of the network between users and the host. If the baseline increases, alarms are used to notify that the network latency is becoming an issue.

When you are establishing an application benchmark, pay close attention to both network and transaction latencies and assess whether or not they appear across a wide range of users (especially in different locations) and/or a wide range of applications. During your analysis, you're looking for primary events – anything related to "slow." Depending on the events you see, you'll know who is at fault. For example, HTTP slow response time, Oracle slow response time, or inefficient client are typical application events. Whereas, TCP slow segment recovery, retransmissions, fragmentation, slow acknowledgements, or low throughput, are typical network events.



```
Packet  Relative Time  Delta Time  PacketVisualizer  Summary                                                              Expert
1063    0.039047       0.006624                      IP L=  400  TCP .AP... S=1729322141     L=  360   102886598...
1064    0.075009       0.035962                      IP L=   40  TCP .A....   1729322501=A   L=    0 S=102886598...
1065    0.075823       0.000814                      IP L=  166  TCP .AP...   1729322501=A   L=  126 S=102886598...
1066    0.077372       0.001540                      IP L= 1068  TCP .AP... S=1729322501     L= 1028   102886610...
1067    0.156410       0.079038                      IP L=   40  TCP .A....   1729323529=A   L=    0 S=102886610
                                   0.100000
                                   0.200000
                                   0.300000
                                   0.400000
                                   0.500000
                                   0.600000
                                   0.700000
                                   0.800000
1084    1.010673       0.854263                      IP L= 1300  TCP .A....   1729323529=A   L= 1260 S=10288661                  conds from packet 1,066)
1085    1.011052       0.000379                      IP L=  205  TCP .AP...   1729323529=A   L=  165 S=102886736...
1086    1.011096       0.000044                      IP L=   40  TCP .A.... S=1729323529     L=    0   102886753...
1087    1.013366       0.002270                      IP L= 1300  TCP .A....   1729323529=A   L= 1260 S=102886753...
1088    1.049019       0.035653                      IP L= 1300  TCP .A....   1729323529=A   L= 1260 S=102886879...
1089    1.049071       0.000052                      IP L=   40  TCP .A.... S=1729323529     L=    0   102887005...
1090    1.050451       0.001380                      IP L= 1300  TCP .A....   1729323529=A   L= 1260 S=102887005...
1091    1.050519       0.000068                      IP L=   40  TCP .A.... S=1729323529     L=    0   102887131...
1092    1.052257       0.001738                      IP L=  533  TCP .AP...   1729323529=A   L=  493 S=102887131...
1094    1.054777       0.002520                      IP L=  955  TCP .AP... S=17                         7180...
1096    1.095381       0.040604                      IP L=   40  TCP .A....   172                        7180...
                                   0.100000
                                   0.200000
                                   0.300000
1097    1.465195       0.369814                      IP L= 1300  TCP .A....   1729324444=A   L= 1260 S=102887180...  Slow Server Response Time (0.410418 seconds from packet 1,094)
1098    1.466141       0.000946                      IP L=  205  TCP .AP...   1729324444=A   L=  165 S=102887306...
1099    1.466208       0.000067                      IP L=   40  TCP .A.... S=1729324444     L=    0   102887323...
1100    1.469387       0.003179                      IP L= 1300  TCP .A....   1729324444=A   L= 1260 S=102887323...
1101    1.470305       0.000918                      IP L=  205  TCP .AP...   1729324444=A   L=  165 S=102887449...
```

Latency monitoring can help correlate areas of latency with other relevant statistics, as well as the actual network traffic occurring at that time. This type of high-resolution forensic analysis can help to detect latency problems at the highest level and drill down quickly for a closer look.

Latency monitors can also include a feature that sets thresholds on latency, so alarms will go off when normal conditions are exceeded. You can be made aware of excessive latency before application performance becomes a widespread issue, allowing you to make necessary adjustments to the network proactively. This type of proactive latency monitoring allows you to detect and correct problems in the network and applications before users even notice a slowdown.

After receiving that first notification, and before diving into packet analysis head first, enterprise-grade network analysis solutions also typically include some level of “expert” analysis, or background analysis that indicates possible sources of performance issues, like slow server response time or too many packet retransmissions. Expert analysis can often guide you to the exact cause of the problem with just a few mouse clicks.

Importance of Application Response Time Assessment in Cloud Computing

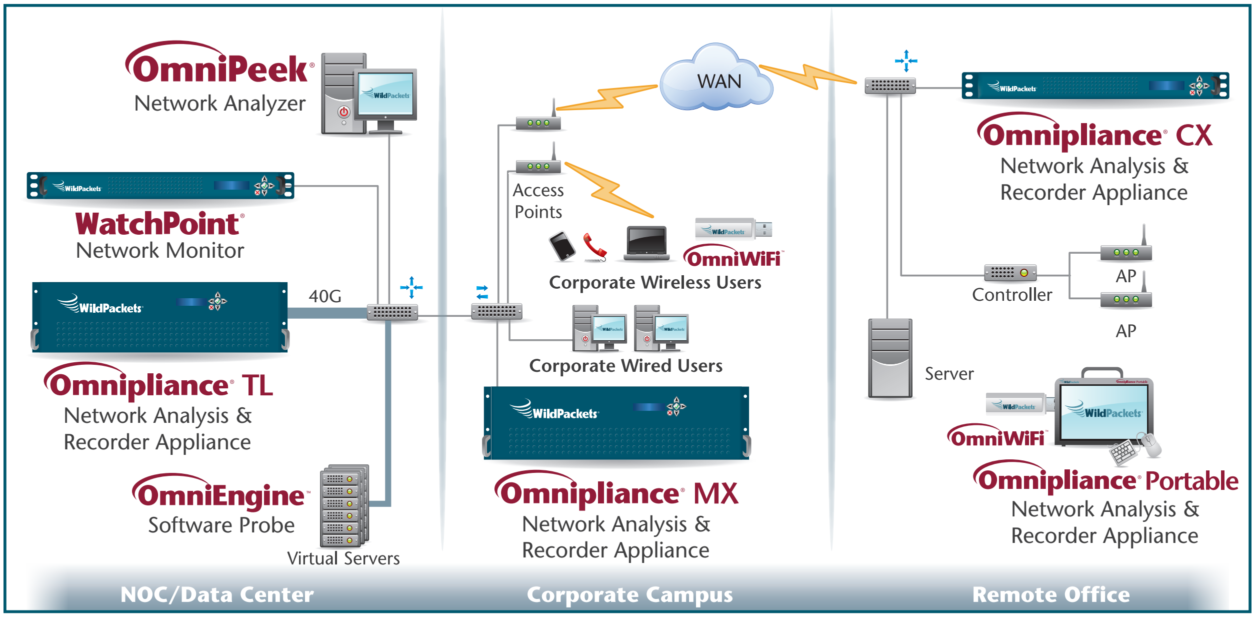
What’s missing in most cloud computing discussions is how cloud computing affects the monitoring and management of your network. From a network perspective, whether your server is located in your data center or in the cloud, the only thing that’s changed is that the blame has shifted; you still need to monitor and analyze your network traffic. Additionally, when you discover problems occurring on the application side, you’ll now have to deal with multiple external vendors.

24x7 Monitoring and Troubleshooting Distributed Application Performance

Remember that shifting application servers to another location does not always make the most sense, and before you make the shift, keep latency in mind. Latency typically increases when you move to the cloud because the distance to application and data servers can increase greatly, along with the number of hops for data packets. And with the move to the cloud, you lose control in improving latency.

Once your applications are in the cloud, monitoring and analyzing your network will help you compare against performance claims from cloud vendors. You can verify whether your cloud vendor is living up to their service level agreements, and act as a sort of 'watchdog' if they are not. You'll be shifting from managing your own infrastructure to managing service availability and performance.

WildPackets Distributed Network Analysis Solutions



In order to make sure your applications are performing properly, you need to first determine what optimal performance is, and have tools in place that can perform 24/7 monitoring on your applications. With WildPackets Distributed Network Analysis Solutions you can monitor application response time, round-trip network delay, server responsiveness, database transactions per second, and a myriad of other low-level statistics. You can also use the Application Performance Index to standardize the overall assessment of user satisfaction.

A key factor in solving application performance issues is having the ability to analyze down to the packet level. However, having only a packet-level view is not enough. You need a distributed network analysis solution that provides both high-level monitoring capabilities that can keep you aware of how your applications are functioning from a business perspective, and that performs deep packet analysis when a problem does occur.

WatchPoint network monitor and OmniPeek Enterprise network analyzer, in conjunction with Omnipliance network analysis and recorder appliances, deliver high-level monitoring capabilities with one-click packet drill down while supporting both local and remote data capture and analysis—the perfect solution for distributed network analysis.

24x7 access to ALL network data lets you:

- Access and analyze real-time data vs. recreating issues post-case
- Easily share data, bringing problems to resolution more quickly
- Reduce and can even eliminate the need for “fly and fix” network engineers every time a problem occurs at a remote office

With WildPackets Distributed Network Analysis Solutions, you’ll have vital information for managing and troubleshooting your entire network at your fingertips, 24x7.

Conclusion

Geographically distributed companies can no longer deal with network issues “the old-fashioned” way – sending network engineers to physically visit a site with dedicated test hardware, generating data that can only be analyzed locally. This is too expensive, far too time consuming, and functionally limits the analysis capability to a single individual.

Organizations of all sizes are geographically distributed today. They require a distributed network management and analysis solutions that can seamlessly extend to remote locations so network administrators can monitor, analyze, and troubleshoot their entire network without having to leave their office.

No matter what solution you select for analyzing how new and existing applications behave on a network, it must meet the following criteria:

1. Provide detailed information through all seven layers of the OSI model.
2. Enable collection of key network data, throughout the entire infrastructure extending to remote locations, over meaningful time cycles (like a week) to develop high quality network baseline data.
3. Quickly identify if the network is being stressed and where.
4. Offer “What If” capabilities for analyzing the impact of new developments and implementations.

About WildPackets, Inc.

WildPackets develops hardware and software solutions that drive network performance, enabling organizations of all sizes to analyze, troubleshoot, optimize, and secure their wired and wireless networks. WildPackets products are sold in over 60 countries and deployed in all industrial sectors. Customers include Boeing, Chrysler, Motorola, Nationwide, and over 80 percent of the Fortune 1000. WildPackets is a Cisco Technical Development Partner (CTDP).

To learn more about WildPackets solutions, please visit www.wildpackets.com, or contact WildPackets Sales: sales@ wildpackets.com or (925) 937-3200.